## 一、创建应用

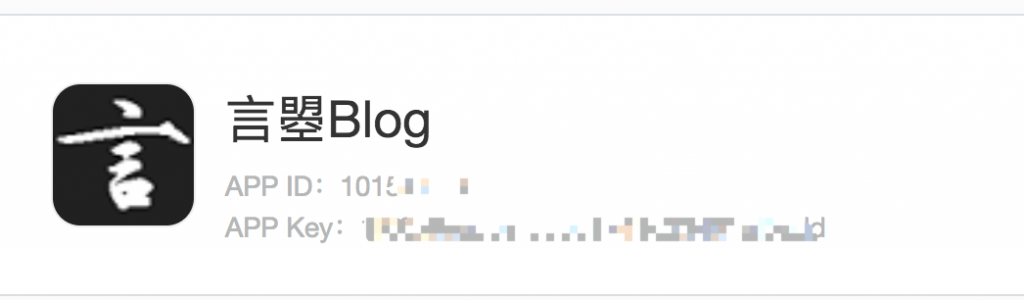
**1、在 QQ互联 创建应用**

地址：https://connect.qq.com/manage.html#/

然后进行实名认证，创建应用，审核通过

[](https://liuyanzhao.com/wp-content/uploads/2018/05/QQ20190120-204901@2x.png)

然后点击查看，可以获得 APP ID 和 APP Key

[](https://liuyanzhao.com/wp-content/uploads/2018/05/QQ20190120-204945@2x.png)

回调地址如下

[](https://liuyanzhao.com/wp-content/uploads/2018/05/QQ20190120-205011@2x.png)

**2、授权的基本原理**

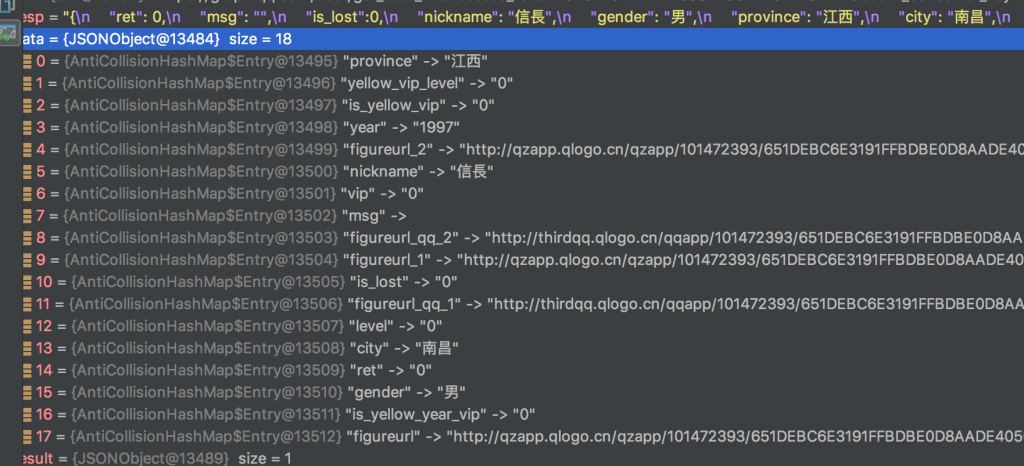
可以参考[官方文档](https://liuyanzhao.com/wp-content/themes/begin/inc/go.php?url=http://wiki.connect.qq.com/%E5%87%86%E5%A4%87%E5%B7%A5%E4%BD%9C_oauth2-0" \t "https://liuyanzhao.com/_blank)

1）根据QQ登录链接可以回调获得 code

2）根据APP ID 、APP Key 和 code 可获得 token

3）根据 token 获得 OpenId

4)  根据 OpenId 可以获得用户的基本信息



其中 OpenId 是一个唯一的值，比如 8A674574E1B12345D790A111EFE81234

**3、几个必要的URL**

1）登录页面授权 URL：

https://graph.qq.com/oauth2.0/authorize?response\_type=code&client\_id=%s&redirect\_uri=%s&scope=%s

2）获得 Token 的 URL：

https://graph.qq.com/oauth2.0/token?grant\_type=authorization\_code&client\_id=%s&client\_secret=%s&code=%s&redirect\_uri=%s

3）获得用户OpenId 的 URL：

https://graph.qq.com/oauth2.0/me?access\_token=%s

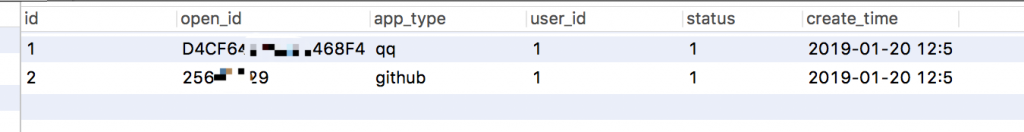
4）获得用户信息的 URL：

https://graph.qq.com/user/get\_user\_info?access\_token=%s&oauth\_consumer\_key=%s&openid=%s

## 二、基本准备

**1、数据库设计**

直接看图片

[](https://liuyanzhao.com/wp-content/uploads/2018/05/QQ20190120-204221@2x.png)

**2、放置 QQ 登录按钮**

这里有两种常用的方式

① 弹小窗

1. <script>
2. function openWin(url,name,iWidth,iHeight) {
3. //获得窗口的垂直位置
4. var iTop = (window.screen.availHeight - 30 - iHeight) / 2;
5. //获得窗口的水平位置
6. var iLeft = (window.screen.availWidth - 10 - iWidth) / 2;
7. window.open(url, name, 'height=' + iHeight + ',innerHeight=' + iHeight + ',width=' + iWidth + ',innerWidth=' + iWidth + ',top=' + iTop + ',left=' + iLeft + ',status=no,toolbar=no,menubar=no,location=no,resizable=no,scrollbars=0,titlebar=no');
8. }
9. function qqLogin() {
10. var url = "https://graph.qq.com/oauth2.0/authorize?response\_type=code&client\_id=11111111&redirect\_uri=http://localhost:8080/oauth/qq/callback&scope=get\_user\_info";
11. openWin(url,"qqLogin",650,500);
12. }
13. </script>
14. <a  href="javascript:void(0);" onclick="qqLogin()"></a>



② 在新窗口打开

1. **<a** href="https://graph.qq.com/oauth2.0/authorizeresponse\_type=code&client\_id=11111111&redirect\_uri=http://localhost:8080/oauth/qq/callback&scope=get\_user\_info" target="\_blank"**></a>**

## 三、具体代码

参考[这里](https://liuyanzhao.com/wp-content/themes/begin/inc/go.php?url=https://blog.csdn.net/u010978008/article/details/73873868" \t "https://liuyanzhao.com/_blank)

1、由于做了多个登录，所以代码做了一定程度的封装，大致如下：

1. **package** com.liuyanzhao.sens.service;
2. **import** com.liuyanzhao.sens.model.dto.BindUserDTO;
3. **import** com.liuyanzhao.sens.model.dto.JsonResult;
4. **import** com.liuyanzhao.sens.utils.Response;
5. **import** java.io.UnsupportedEncodingException;
6. /\*\*
7. \* @author 言曌
8. \* @date 2019/1/20 上午10:24
9. \*/
10. **public** **interface** AuthService {
11. /\*\*
12. \* 根据code获得Token
13. \*
14. \* @param code code
15. \* @return token
16. \*/
17. Response<String> getAccessToken(String code);
18. /\*\*
19. \* 根据Token获得OpenId
20. \*
21. \* @param accessToken Token
22. \* @return openId
23. \*/
24. Response<String> getOpenId(String accessToken);
25. /\*\*
26. \* 刷新Token
27. \*
28. \* @param code code
29. \* @return 新的token
30. \*/
31. Response<String> refreshToken(String code);
32. /\*\*
33. \* 拼接授权URL
34. \*
35. \* @return URL
36. \*/
37. Response<String> getAuthorizationUrl();
38. /\*\*
39. \* 根据Token和OpenId获得用户信息
40. \*
41. \* @param accessToken Token
42. \* @param openId openId
43. \* @return 第三方应用给的用户信息
44. \*/
45. Response<BindUserDTO> getUserInfo(String accessToken, String openId);
46. }

2、由于全部是自己封装的，所以http请求的代码也是所有的登录共用的，这里统一放放到了类DefaultAuthServiceImpl中，代码如下：

1. **package** com.liuyanzhao.sens.service.impl;
2. **import** com.liuyanzhao.sens.service.AuthService;
3. **import** org.springframework.http.client.SimpleClientHttpRequestFactory;
4. **import** org.springframework.http.converter.ByteArrayHttpMessageConverter;
5. **import** org.springframework.http.converter.HttpMessageConverter;
6. **import** org.springframework.http.converter.ResourceHttpMessageConverter;
7. **import** org.springframework.http.converter.StringHttpMessageConverter;
8. **import** org.springframework.http.converter.json.MappingJackson2HttpMessageConverter;
9. **import** org.springframework.http.converter.support.AllEncompassingFormHttpMessageConverter;
10. **import** org.springframework.http.converter.xml.SourceHttpMessageConverter;
11. **import** org.springframework.web.client.RestTemplate;
12. **import** javax.xml.transform.Source;
13. **import** java.nio.charset.StandardCharsets;
14. **import** java.util.LinkedList;
15. **import** java.util.List;
16. /\*\*
17. \* @author 言曌
18. \* @date 2018/5/9 下午3:02
19. \*/
20. **public** **abstract** **class** DefaultAuthServiceImpl **implements** AuthService {
21. **public** **static** RestTemplate getRestTemplate() {// 手动添加
22. SimpleClientHttpRequestFactory requestFactory=**new** SimpleClientHttpRequestFactory();
23. requestFactory.setReadTimeout(120000);
24. List<HttpMessageConverter<?>> messageConverters = **new** LinkedList<>();
25. messageConverters.add(**new** ByteArrayHttpMessageConverter());
26. messageConverters.add(**new** StringHttpMessageConverter(StandardCharsets.UTF\_8));
27. messageConverters.add(**new** ResourceHttpMessageConverter());
28. messageConverters.add(**new** SourceHttpMessageConverter<Source>());
29. messageConverters.add(**new** AllEncompassingFormHttpMessageConverter());
30. messageConverters.add(**new** MappingJackson2HttpMessageConverter());
31. RestTemplate restTemplate=**new** RestTemplate(messageConverters);
32. restTemplate.setRequestFactory(requestFactory);
33. **return** restTemplate;
34. }
35. }

由此，所有的登录Service只需要继承AuthService即可。

3、QQ登录 Service 接口

QQAuthService.java

1. **package** com.liuyanzhao.sens.service;
2. /\*\*
3. \* @author 言曌
4. \* @date 2018/5/9 下午3:00
5. \*/
6. **public** **interface** QQAuthService **extends** AuthService {
7. }

4、QQ登录 Service 实现

QQAuthServiceImpl.java

1. **package** com.liuyanzhao.sens.service.impl;
2. **import** com.alibaba.fastjson.JSONObject;
3. **import** com.liuyanzhao.sens.model.dto.BindUserDTO;
4. **import** com.liuyanzhao.sens.service.QQAuthService;
5. **import** com.liuyanzhao.sens.utils.Response;
6. **import** lombok.extern.slf4j.Slf4j;
7. **import** org.springframework.stereotype.Service;
8. **import** org.springframework.web.util.UriComponentsBuilder;
9. **import** java.net.URI;
10. **import** java.util.HashMap;
11. **import** java.util.Map;
12. /\*\*
13. \* @author 言曌
14. \* @date 2018/5/9 下午3:15
15. \*/
16. @Service
17. @Slf4j
18. **public** **class** QQAuthServiceImpl **extends** DefaultAuthServiceImpl **implements** QQAuthService {
19. //QQ 登陆页面的URL
20. **private** **final** **static** String AUTHORIZATION\_URL =
21. "https://graph.qq.com/oauth2.0/authorize?response\_type=code&client\_id=%s&redirect\_uri=%s&scope=%s";
22. //获取token的URL
23. **private** **final** **static** String ACCESS\_TOKEN\_URL = "https://graph.qq.com/oauth2.0/token?grant\_type=authorization\_code&client\_id=%s&client\_secret=%s&code=%s&redirect\_uri=%s";
24. // 获取用户 openid 的 URL
25. **private** **static** **final** String OPEN\_ID\_URL = "https://graph.qq.com/oauth2.0/me?access\_token=%s";
26. // 获取用户信息的 URL，oauth\_consumer\_key 为 apiKey
27. **private** **static** **final** String USER\_INFO\_URL = "https://graph.qq.com/user/get\_user\_info?access\_token=%s&oauth\_consumer\_key=%s&openid=%s";
28. // 下面的属性可以通过配置读取
29. // QQ 在登陆成功后回调的 URL，这个 URL 必须在 QQ 互联里填写过
30. **private** **static** **final** String CALLBACK\_URL = "http://localhost:8080/oauth/qq/callback";
31. // QQ 互联应用管理中心的 APP ID
32. **private** **static** **final** String APP\_ID = "111111111";
33. // QQ 互联应用管理中心的 APP Key
34. **private** **static** **final** String APP\_SECRET = "111111111111111111111111111111111";
35. // QQ 互联的 API 接口，访问用户资料
36. **private** **static** **final** String SCOPE = "get\_user\_info";
37. @Override
38. **public** Response<String> getAuthorizationUrl() {
39. String url = String.format(AUTHORIZATION\_URL, APP\_ID, CALLBACK\_URL, SCOPE);
40. **return** Response.yes(url);
41. }
42. @Override
43. **public** Response<String> getAccessToken(String code) {
44. String url = String.format(ACCESS\_TOKEN\_URL, APP\_ID, APP\_SECRET, code, CALLBACK\_URL);
45. UriComponentsBuilder builder = UriComponentsBuilder.fromHttpUrl(url);
46. URI uri = builder.build().encode().toUri();
47. String resp = getRestTemplate().getForObject(uri, String.**class**);
48. **if** (resp != **null** && resp.contains("access\_token")) {
49. Map<String, String> map = getParam(resp);
50. String access\_token = map.get("access\_token");
51. **return** Response.yes(access\_token);
52. }
53. log.error("QQ获得access\_token失败，code无效，resp:{}", resp);
54. **return** Response.no("code无效！");
55. }
56. //由于QQ的几个接口返回类型不一样，此处是获取key-value类型的参数
57. **private** Map<String, String> getParam(String string) {
58. Map<String, String> map = **new** HashMap();
59. String[] kvArray = string.split("&");
60. **for** (**int** i = 0; i < kvArray.length; i++) {
61. String[] kv = kvArray[i].split("=");
62. map.put(kv[0], kv[1]);
63. }
64. **return** map;
65. }
66. //QQ接口返回类型是text/plain，此处将其转为json
67. **private** JSONObject ConvertToJson(String string) {
68. string = string.substring(string.indexOf("(") + 1, string.length());
69. string = string.substring(0, string.indexOf(")"));
70. JSONObject jsonObject = JSONObject.parseObject(string);
71. **return** jsonObject;
72. }
73. @Override
74. **public** Response<String> getOpenId(String accessToken) {
75. String url = String.format(OPEN\_ID\_URL, accessToken);
76. UriComponentsBuilder builder = UriComponentsBuilder.fromHttpUrl(url);
77. URI uri = builder.build().encode().toUri();
78. String resp = getRestTemplate().getForObject(uri, String.**class**);
79. **if** (resp != **null** && resp.contains("openid")) {
80. JSONObject jsonObject = ConvertToJson(resp);
81. String openid = jsonObject.getString("openid");
82. **return** Response.yes(openid);
83. }
84. log.error("QQ获得openid失败，accessToken无效，resp:{}", resp);
85. **return** Response.no(resp);
86. }
87. @Override
88. **public** Response<BindUserDTO> getUserInfo(String accessToken, String openId) {
89. String url = String.format(USER\_INFO\_URL, accessToken, APP\_ID, openId);
90. UriComponentsBuilder builder = UriComponentsBuilder.fromHttpUrl(url);
91. URI uri = builder.build().encode().toUri();
92. String resp = getRestTemplate().getForObject(uri, String.**class**);
93. JSONObject data = JSONObject.parseObject(resp);
94. BindUserDTO result = **new** BindUserDTO();
95. result.setOpenId(openId);
96. result.setGender(data.getString("gender"));
97. result.setAvatar(data.getString("figureurl\_qq\_2"));
98. result.setNickname(data.getString("nickname"));
99. **return** Response.yes(result);
100. }
101. @Override
102. **public** Response<String> refreshToken(String code) {
103. **return** **null**;
104. }
105. }

5、在Controller中调用，代码如下：

1. **package** com.liuyanzhao.sens.web.controller.front;
2. **import** cn.hutool.core.date.DateUtil;
3. **import** cn.hutool.extra.servlet.ServletUtil;
4. **import** com.liuyanzhao.sens.entity.Log;
5. **import** com.liuyanzhao.sens.entity.ThirdAppBind;
6. **import** com.liuyanzhao.sens.entity.User;
7. **import** com.liuyanzhao.sens.model.dto.LogsRecord;
8. **import** com.liuyanzhao.sens.model.dto.SensConst;
9. **import** com.liuyanzhao.sens.model.enums.BindTypeEnum;
10. **import** com.liuyanzhao.sens.service.\*;
11. **import** com.liuyanzhao.sens.utils.Response;
12. **import** lombok.extern.slf4j.Slf4j;
13. **import** org.springframework.beans.factory.annotation.Autowired;
14. **import** org.springframework.stereotype.Controller;
15. **import** org.springframework.web.bind.annotation.GetMapping;
16. **import** org.springframework.web.bind.annotation.RequestParam;
17. **import** javax.servlet.http.HttpServletRequest;
18. /\*\*
19. \* @author 言曌
20. \* @date 2018/5/9 下午2:59
21. \*/
22. @Controller
23. @Slf4j
24. **public** **class** AuthController {
25. @Autowired
26. **private** QQAuthService qqAuthService;
27. @Autowired
28. **private** UserService userService;
29. @Autowired
30. **private** ThirdAppBindService thirdAppBindService;
31. @Autowired
32. **private** LogService logService;
33. /\*\*
34. \* 第三方授权后会回调此方法，并将code传过来
35. \*
36. \* @param code code
37. \* @return
38. \*/
39. @GetMapping("/oauth/qq/callback")
40. **public** String oauthByQQ(@RequestParam(value = "code") String code, HttpServletRequest request) {
41. Response<String> tokenResponse = qqAuthService.getAccessToken(code);
42. **if** (tokenResponse.isSuccess()) {
43. Response<String> openidResponse = qqAuthService.getOpenId(tokenResponse.getData());
44. **if** (openidResponse.isSuccess()) {
45. //根据openId去找关联的用户
46. ThirdAppBind bind = thirdAppBindService.findByAppTypeAndOpenId(BindTypeEnum.QQ.getValue(), openidResponse.getData());
47. **if** (bind != **null** && bind.getUserId() != **null**) {
48. //执行Login操作
49. User user = userService.findByUserId(bind.getUserId());
50. **if** (user != **null**) {
51. request.getSession().setAttribute(SensConst.USER\_SESSION\_KEY, user);
52. logService.saveByLog(**new** Log(LogsRecord.LOGIN, LogsRecord.LOGIN\_SUCCESS+"(QQ登录)", ServletUtil.getClientIP(request), DateUtil.date()));
53. log.info("用户[{}]登录成功(QQ登录)。", user.getUserDisplayName());
54. **return** "redirect:/admin";
55. }
56. }
57. }
58. }
59. **return** "redirect:/admin/login";
60. }
61. }

查看更多

[SpringBoot网站添加第三方登录之GitHub登录](https://liuyanzhao.com/8223.html" \t "https://liuyanzhao.com/_blank)